RENAME

Vulnerable to TOCTOU issues

Sean Barnum, Cigital, Inc. [vita[1]]

2007-04-04

Part "Original Cigital Coding Rule in XML"

Mime-type: text/xml, size: 10375 bytes

Attack Category	• Path spoofing or confusion problem
Vulnerability Category	• Indeterminate File/Path • TOCTOU - Time of Check, Time of Use
Software Context	• File Management • Filename Management
Location	• stdio.h
Description	The rename() function changes the name of a file. The old argument points to the pathname of the file to be renamed. The new argument points to the new pathname of the file. If old and new both refer to the same existing file, the rename() function returns successfully and performs no other action. If old points to the pathname of a file that is not a directory, new must not point to the pathname of a directory. If the link named by new exists, it will be removed and old will be renamed to new. In this case, a link named new must remain visible to other processes throughout the renaming operation and will refer to either the directory referred to by new or the directory referred to as old before the operation began. If old points to the pathname of a directory, new must not point to the pathname of a file that is not a directory. If the directory named by new exists, it will be removed and old will be renamed to new. In this case, a link named new will exist throughout the renaming operation and will refer to either the file referred to by new or the file referred to as old before the operation began. Thus, if new names an existing directory, it must be an empty directory. The new pathname must not contain a path prefix that names old. Write access permission is required

[1] http://buildsecurityin.us-cert.gov/bsi-rules/35-BSI.html (Barnum, Sean)

for both the directory containing old and the directory containing new. If old points to the pathname of
a directory, write access permission is required for the directory named by old and, if it exists, the directory named by new.

If the directory containing old has the sticky bit set, at least one of the following conditions must be true:
o the user must own old
o the user must own the directory containing old
o old must be writable by the user

A call to rename() should be flagged if either argument is referenced earlier in a check-category call.

APIs

Function Name	Comments
_rename	use; win32
_trename	use; win32
_wrename	use; win32
rename	use

Method of Attack

The key issue with respect to TOCTOU vulnerabilities is that programs make assumptions about atomicity of actions. It is assumed that checking the state or identity of a targeted resource followed by an action on that resource is all one action. In reality, there is a period of time between the check and the use that allows either an attacker to intentionally or another interleaved process or thread to unintentionally change the state of the targeted resource and yield unexpected and undesired results.

The rename() call is a use-category call, which when preceded by a check-category call can be indicative of a TOCTOU vulnerability.

A TOCTOU attack in regards to rename() can occur when

a. A check for the existence of either filename occurs.

b. A file rename occurs

Between a and b, an attacker could, for example, link either file (old or new) to an attack file, resulting in either an unintended file being renamed or a file being renamed in an unintended fashion.

Exception Criteria

Solutions

Solution Applicability	Solution Description	Solution Efficacy

Generally applies to any rename().	Translate rename() into function(s) using file descriptors.	Effective
This solution is applicable if the program is a setuid or setgid program.	Drop privileges down to those of the user who is executing the program when executing rename(). That way, the any files involved in the rename() are treated just as if the user issued the rename command at the prompt.	The user will not be able to rename any files he or she doesn't already have permissions to rename.
Generally applies to any rename().	The most basic advice for TOCTOU vulnerabilities is to not perform a check before the use. This does not resolve the underlying issue of the execution of a function on a resource whose state and identity cannot be assured, but it does help to limit the false sense of security given by the check. Attempt the rename and check and then check status after the creation.	Does not resolve the underlying vulnerability but limits the false sense of security given by the check. Checking the status after the operation does not change the fact that the operation may have been exploited but it does allow halting of the application in an error state to help limit further damage.
Generally applies to any rename().	Limit the interleaving of operations on files from	Does not eliminate the underlying vulnerability

	multiple processes.	but can help make it more difficult to exploit.
Generally applies to any rename().	Limit the spread of time (cycles) between the check and use of a resource.	Does not eliminate the underlying vulnerability but can help make it more difficult to exploit.
Generally applies to any rename().	Recheck the resource after the use call to verify that the action was taken appropriately.	Effective in some cases.

Signature Details	int rename(const char *old, const char *new);

Examples of Incorrect Code	`/* rename here will be executed`
	`with the privileges of the process`
	`*/`
	`rename("old.txt", "new.txt");`

Examples of Corrected Code	`//Get the effective user of the`
	`running process. This will be the`
	`program's user or group owner if`
	`setuid or setgid is used.`
	`uid_t init_uid = geteuid();`
	`gid_t init_gid = getegid();`
	`//Drop to the privileges of the`
	`user who is runnig the process.`
	`seteuid(getuid());`
	`setegid(getgid());`
	`if (rename("old.txt", "new.txt")`
	`< 0) //Do the rename with the`
	`privileges of the user running the`
	`process`
	`return -1; //Handle any error`
	`that may occur`

Source References	• Viega, John & McGraw, Gary. *Building Secure Software: How to Avoid Security Problems the Right Way*. Boston, MA: Addison-Wesley Professional, 2001, ISBN: 020172152X , ch 9 • man page for rename() • Microsoft Developer Network Library (MSDN) • http://www.sikurezza.org/ml/06_04/msg00325.html

<table>
<tr><td><b>Recommended Resource</b></td><td></td></tr>
<tr><td><b>Discriminant Set</b></td><td><table>
<tr><td><b>Operating Systems</b></td><td>• UNIX
• Windows</td></tr>
<tr><td><b>Language</b></td><td></td></tr>
</table></td></tr>
</table>

Cigital, Inc. Copyright

The Build Security In (BSI) portal is sponsored by the U.S. Department of Homeland Security (DHS), National Cyber Security Division. The Software Engineering Institute (SEI) develops and operates BSI. DHS funding supports the publishing of all site content.

[1] mailto:copyright@cigital.com